

AgentSQL

Zhidun

MCP Integration Guide

v4.6.0

agentsql.cn

v4.6.0 Documentation

Contents

- v3.9.0 Oracle and SQL Server controlled reads
- v3.8.0 native database tools
- Select transfer method
  - Streamable HTTP: preferred when the service is already running
  - Optional personnel identity delegation (from v1.0.0, extended to v1.1.0)
  - 2026-07-28 Stateless HTTP (from v0.7.0)
  - v1.2.0 Optional MCP OAuth entry
  - stdio: stand-alone desktop agent
- MCP tools
- Call timing
  - read-only query
  - Write operation: automatically transferred to approval
  - Write operation: actively submit for review
- Correctly handle return values
- Claude Desktop access
  - Windows Claude Desktop: Connect to an existing AgentSQL service
- MCP Inspector local acceptance (Streamable HTTP)
- Dify self-hosted access (Streamable HTTP)
- Cursor access (stdio)
- Doubao and other remote MCP clients (Streamable HTTP)
- Find from MCP Registry / Glama
- Client access security reminder
- Self-developed client
  - initialize naked HTTP example
  - Go SDK
- Troubleshooting
  - 401 unauthorized
  - stdio has no console UI
  - Can only be accessed locally on the server

MCP handshake failed

Stacked multiple statements are rejected

Write operation prompt is missing reason

stdio cannot connect to the control plane or cannot decrypt the data source

SQLBot

AgentSQL MCP Access Guide

Language: English | Simplified Chinese

> The current official release is v4.6.0. Start with three usage modes to choose an independent MCP gateway or combined use with native Q&A, then verify the official assets, signatures, and acceptance limits in the v4.6.0 Release. The remaining versioned sections retain the specific conditions of their historical capabilities.

This article explains how to connect the MCP client to AgentSQL. For identity, authorization, rules, approval and capability boundaries, see the user manual . For first-time deployment, see for quick start . For outsourcing decision-making events to Webhook, alarm groups or Syslog, see notification outgoing guide .

In the current version, `list_datasources` returns a `support` declaration for data sources visible to the Agent, including the support tier, available tools, scope, and evidence documentation. Every request still goes through authorization. After seeding an isolated environment with the repository's demo manifest, run `scripts/mcp-acceptance.py` to check tool discovery, an allowed query, an R010 denial, masking, and two durable audit IDs. The API key is read only from `AGENTSQL_MCP_ACCEPTANCE_KEY`:

```
AGENTSQL_MCP_ACCEPTANCE_KEY="$DEMO_AGENT_KEY" python3 scripts/mcp-acceptance.py \
  --mcp-url http://127.0.0.1:7780/mcp --datasource-id ds-demo-pg \
  --allow-sql 'SELECT c.id, c.full_name, c.phone, c.email, c.region, o.status FROM
public.demo_b2_customers c JOIN public.demo_b2_orders o ON o.customer_id=c.id WHERE o.id=1' \
  --deny-sql 'SELECT * FROM public.demo_b2_customers WHERE id = 1' --deny-rule R010
```

This example depends on the column grants in `examples/docker/demo-seed.yaml`. For another environment, choose an authorized query and a query rejected by R010. Do not send real customer data to the public demo. The script prints only the decision, audit IDs, and masked-cell count.

v3.9.0 Oracle and SQL Server controlled reads

- For specified Oracle 26ai Free, choose `verify-full`, supply an absolute CA file path, and use a connection host matching the certificate name. The gateway verifies the certificate on every new connection. Oracle wallets, mTLS and RAC remain outside this acceptance scope.
- Enable `{"oracle":{"explain":true}}` in `read_control_json` only with a dedicated non-system readonly account and a verified `PLAN_TABLE`. Authorized narrow SELECTs then receive bounded plan checks; unavailable or invalid plans fail closed. Existing Oracle sources keep plan checks disabled by default.

- Use MCP `query` for both Oracle and SQL Server. On SQL Server, keep strict certificate validation and scoped SELECT/SHOWPLAN grants. Validate allowed and denied calls, masking and durable audit IDs after certificate rotation and recovery. The public Demo remains synthetic PostgreSQL/MySQL. See the release notes, upgrade guide and acceptance record.

v3.8.0 native database tools

- `find_documents` accepts `datasource_id`, `collection`, explicit top-level `fields`, string equality `filter`, and a `limit` of 1–100. Replica set mode requires the filter field set to match an administrator-approved index. Aggregation, arbitrary commands and writes are unavailable.
- `read_hash_fields` accepts `datasource_id`, an exact `key`, and 1–16 explicit `fields`. Each Hash field needs a separate grant. The response is capped at 64 KiB, masked by field source and delivered only after durable terminal audit; see the v3.7.0 acceptance for topology evidence.
- `query_analytics` can tighten query scan budgets through administrator `read_control_json` on the specified ClickHouse 26.8.31.2 single node. The readonly database role must permit constrained per-query settings; without a profile the prior budget applies.
- `search_vectors` can enable authoritative tenant scope through administrator `read_control_json` on the specified Milvus 2.6.25 standalone instance. Optional `filter` accepts at most four approved static string field equalities. The current Agent supplies the tenant value, never the client. The public Demo still shows only synthetic PostgreSQL/MySQL scenarios; see the v3.8.0 design and acceptance.

Select transfer method

Streamable HTTP: preferred when the service is already running

When the installer, systemd or Compose is already running AgentSQL, connect directly:

- Endpoint: `http://127.0.0.1:7780/mcp`
- Identity header: `Authorization: Bearer <Agent API Key>`
- MCP protocol version: `2025-06-18` or `2025-11-25` (both are old handshake protocols; the new version uses the following independent entrance)
- `Accept`: `application/json, text/event-stream`
- Mode: Default stateful (initialize returns `Mcp-Session-Id`); can explicitly fall back to stateless
- Disconnection replay: The old stateful protocol persists SSE events by default; reconnect with `GET /mcp` + `Last-Event-ID`. If the event has been recycled due to TTL/capacity, HTTP 400 will be returned, and some events will not be returned.
- Single request body limit: 4 MiB

Agent API Key is not a console administrator token. After the Agent is disabled, expires, deletes or rotates the Key, the old Key becomes invalid immediately.

Optional personnel identity delegation (from v1.0.0, extended to v1.1.0)

Administrators can enable `mcp.delegation.require_human_token: true` in the configuration while maintaining `server.console_enabled: true`. After enabling, each request of the two HTTP MCP portals must carry the Agent Key and the human delegation token respectively:

```
Authorization: Bearer <Agent API Key>
X-AgentSQL-User-Authorization: Bearer <人员委托令牌>
```

Personnel with `query.execute` authority first log in through the console (local account, LDAP or OIDC), and then issue by Agent ID in "Task Authorization → MCP User Identity Delegation"; they can also be called with a console login token:

```
curl -fsS -X POST http://127.0.0.1:7780/api/v1/auth/delegation-token \
  -H "Authorization: Bearer ${AGENTSQL_ADMIN_TOKEN}" \
  -H 'Content-Type: application/json' \
  --data '{"agent_id":"<Agent ID>"}'
```

`data.token` in the response is only applicable to the specified Agent, and cannot access the console API for up to 30 minutes; ordinary console login tokens cannot be directly filled in the second MCP request header. When the original login session exits or expires, the user is deactivated, or `query.execute` is revoked, the delegation token becomes invalid immediately. The stateful `/mcp` session is bound to the person's identity, and replacement requires `initialize` again. The task-level Key also requires that the user to whom the token belongs is consistent with the authorized issuer. Token issuance requires management auditing to be writable, otherwise it fails to close.

Starting from v1.1.0, approval application and execution, and B5 cross-request sessions are also bound to the initiator, agent, and tenant; cross-person reuse or when the original personnel's permissions expire are rejected. Please confirm that the MCP client can safely configure two HTTP identity headers before enabling this switch. Public Live Demos do not use this mode.

Process restart will not restore go-sdk's memory transport session. Requests carrying the old `Mcp-Session-Id` will receive an HTTP 404 with `Mcp-Session-Expired: 1`; the client should discard the old ID and retry `initialize`. The new session will not read events from the old session.

2026-07-28 Stateless HTTP (from v0.7.0)

The new protocol uses `http://127.0.0.1:7780/mcp/2026-07-28`. This path only accepts `2026-07-28` and will not negotiate to the old version; the behavior and default configuration of the old `/mcp` remain unchanged. Each POST of the new path re-verifies the Agent API Key, determines the tenant by this key, and shares the same Agent's QPS limit with the old path. Request bodies are still limited to 4 MiB, and responses continue to adhere to gateway caps.

The client must send every request `Authorization: Bearer <Agent API Key>`、`MCP-Protocol-Version: 2026-07-28`、`Mcp-Method`、`Content-Type: application/json` with `Accept: application/json, text/event-stream`;When calling the tool, you also need to send the `params.name` consistent `Mcp-Name`。`params._meta` Must contain the same version `io.modelcontextprotocol/protocolVersion` and `io.modelcontextprotocol/clientCapabilities`. The client can first call `server/discover` query capability, you can also call the tool directly; no need `initialize` or `notifications/initialized`. This path has no MCP transport session, independent GET event stream, DELETE close or `Last-Event-ID` replay; receive old `Mcp-Session-Id` or `Last-Event-ID` Ignore them when requesting headers. After the connection is disconnected, it is up to the client to decide whether to try again with a new request ID. See detailsMCP 2026-07-28 HTTP SpecificationandGo SDK Protocol Guide。

The new path registers the 12 basic tools listed below for ordinary Agent Key. **does not currently provide the B5 cross-request logical session and planned transaction tool**. The task-level temporary read-only Key only registers `query`, and queries must still pass temporary authorization and original Agent permissions. The business session of B5 is a different concept from the MCP transmission session; continue to use the old `/mcp` or stdio when B5 is needed. `mcp.http.stateful` only controls the old path and does not change the stateless semantics of the new path. The client shall not regard the administrator token or self-reported tenant header as the Agent identity.

Starting from v1.6.0, this stateless portal will pass the HTTP client disconnect to the executing read-only query; the gateway will check again for timeout or cancellation status before delivering the results. The old state `/mcp` continues to terminate the request according to the server statement timeout to preserve the B5 session rollback constraints. If the client tries again after disconnecting, it should use a new request ID and confirm the final status of the previous request according to the audit record; do not regard the disconnection as the query has been successful.

```
curl -fsS http://127.0.0.1:7780/mcp/2026-07-28 \
  -H "Authorization: Bearer ${AGENTSQL_API_KEY}" \
  -H 'Content-Type: application/json' \
  -H 'Accept: application/json, text/event-stream' \
  -H 'MCP-Protocol-Version: 2026-07-28' \
  -H 'Mcp-Method: server/discover' \
  --data '{"jsonrpc":"2.0","id":1,"method":"server/discover","params":{"_meta":
{"io.modelcontextprotocol/protocolVersion":"2026-07-28","io.modelcontextprotocol/clientCapabilities":
{},"io.modelcontextprotocol/clientInfo":{"name":"agentsql-smoke","version":"1.2.0"}}}}'
```

v1.2.0 Optional MCP OAuth entry

`/mcp/oauth` is an independent MCP `2026-07-28` Protected resource entry, closed by default. Before enabling, configure the console OIDC identity source and create the corresponding managed OIDC user in the same tenant; `mcp.oauth.issuer_url` and `tenant_id` Must match the console OIDC configuration exactly. The authorization server must issue an RS256 JWT Access Token to the resource, which `aud` Include**precise** `resource_url`, scope contains a single configured `required_scope`. Production issuer and resource URLs must use HTTPS; HTTP is only used for native loopback testing.

```
mcp:
  oauth:
    enabled: true
    issuer_url: "https://id.example.com/realms/agentsql"
    resource_url: "https://mcp.example.com/mcp/oauth"
    tenant_id: "tenant_default"
    required_scope: "agentsql:mcp"
```

The client is pre-registered in the authorization server and is responsible for the authorization code, PKCE and `resource` parameters. `mcp.json` of MCP Inspector 2.11.0 can use the following form; where `scopes` must be a string, and Agent Key and OAuth Access Token are two independent credentials:

```
{
  "mcpServers": {
    "agentsql-oauth": {
      "type": "http",
      "url": "https://mcp.example.com/mcp/oauth",
      "protocolEra": "modern",
      "headers": {"X-AgentSQL-Agent-Key": "<Agent API Key>"},
      "oauth": {"clientId": "<预注册客户端 ID>", "scopes": "agentsql:mcp"}
    }
  }
}
```

Without Access Token, the server passes `WWW-Authenticate` point to `/.well-known/oauth-protected-resource/mcp/oauth`. User, Agent, Tenant and `query.execute` Permissions are requested to be re-verified each time; console session tokens, old personnel delegation tokens, tokens from other resources and URL query parameters cannot replace the Access Token of this entrance. Currently, only Keycloak 26.4.6 + Inspector 2.11.0 is tested; Dify 1.17.1 only verifies the old version. `/mcp` entrance. See results and boundariesv1.2.0Acceptance record。

The service should only bind to the loopback address by default. When using it remotely, select SSH local forwarding or controlled TLS reverse proxy; prohibit `0.0.0.0:7780` from streaking on the public network.

```
# Applicable premise: the remote AgentSQL only listens to 127.0.0.1:7780; the local machine has
configured SSH identity
ssh -N -L 7780:127.0.0.1:7780 user@agentsql-host
```

Then the client still connects to `http://127.0.0.1:7780/mcp`. TLS, access control, and flow limiting must be configured when using a reverse proxy; do not log the `Authorization` header.

stdio: stand-alone desktop agent

stdio will start a new AgentSQL runtime:

```
# Applicable prerequisite: agentsql has been installed on this machine; the current user can read the
absolute path configuration and control plane storage
AGENTSQL_SECRET='<与控制面配对的恰好 32 字节值>' \
AGENTSQL_API_KEY='<Agent API Key>' \
agentsql mcp --config /absolute/path/config.yaml
```

`--config` can be abbreviated as `-c`, and the default is `config.yaml` in the current directory. Key is provided first through `AGENTSQL_API_KEY`; do not put Key into `--api-key`, otherwise it will appear in the process parameters. stdio and HTTP use the same set of tool registration rules: ordinary Agent Key provides 12 basic tools, and 8 session/transaction tools are provided when B5 is turned on by default; task-level temporary read-only Key only provides `query`.

> **Important: stdio is not a proxy to a running systemd service.** It creates a new set of runtimes that must be able to read configuration, metadata/audit control plane storage, and use the same `AGENTSQL_SECRET` credentials as the encrypted data source. The `/etc/agentsql/agentsql.env` permission generated by the installer is `0600 root:root`, which cannot be read by ordinary desktop users; please use HTTP directly when the service has been installed and running.

stdio mode turns off the console and event streaming; don't expect it to provide a console UI.

MCP tools

All SQL parameters only accept a single statement, stacking of multiple statements is not allowed. The client must read `decision` in the structured result.

Ordinary Agent Key registers the following 12 basic tools; task-level temporary read-only Key only registers `query`:

Tools	Purpose	Add ginseng	Possible decisions and key semantics
`read_key`	Read Redis/Valkey to specify the exact key of a single instance	`datasource_id`、`key`	Only accepts UTF-8 strings authorized by strict TLS/ACL, independent `key` policy; fixed `TYPE`/bounded `GETRANGE`, masked and persistently audited before delivery, does not accept wildcards, native commands, writes or cross-repositories
`find_documents`	Read an explicit top-level scalar field from the specified MongoDB collection	`datasource_id`、`collection`、`fields`、`filter`、`limit`	Only accepts 1–8 string equivalent filter conditions and an upper limit of 1–100 rows; requires accurate set authorization, strict TLS, authenticated accounts, masking before return, and written persistent audit; does not accept aggregation, arbitrary operators or writes
`query_analytics`	Read explicit columns from a specified ClickHouse MergeTree table	`datasource_id`, `table`, `fields`, `filter`, `limit`	Pinned 26.8.31.2 standalone, verified TLS, readonly account, exact table and column grants, String equality filters, and 1–100 output rows. v3.8.0 can configure tighter read budgets. Delivery requires masking and durable audit; native SQL, views, and distributed tables are unavailable.
`search_vectors`	Search one vector in a specified Milvus collection	`datasource_id`, `collection`, `vector_field`, `vector`, `fields`, optional `filter`, `top_k`	Pinned 2.6.25 TLS standalone, `default` database, exact collection and field grants, one finite FloatVector, and 1–20 hits. A v3.8.0 administrator profile can enable authoritative tenant scope and approved-field equality filters. Only masked scalars are returned; native expressions, partitions, writes, and hybrid searches are unavailable.
`list_datasources`	List the data sources that the current Agent has been authorized to use	None	Public identification information is returned successfully, but no host, password or connection parameters are returned; failure is `error`
`list_schema`	Read authorized structure and trim by column permissions	`datasource_id`; `table` optional	Success is usually `allow`; `table` only accepts letters, numbers, underscores, and at most one dot `table` or `schema.table`, quotes and special identifiers are not supported; results exceeding 10,000 rows must be reduced to a specific table

Tools	Purpose	Add ginseng	Possible decisions and key semantics
`explain_query`	Parses and evaluates SQL; dialects with proven planning capabilities ship with execution plans and do not execute queries	`datasource_id`、`sql`	`allow`, `deny`, `warn`, `approve`, or `error` can be returned; even `allow` did not execute the SQL. Oracle Free and YashanDB specify read-only paths and do not pre-execute the plan.
`query`	Execute protected single `SELECT`	`datasource_id`、`sql`	Only query; the results are automatically restricted according to the data source `row_limit` and the result layer is masked; dangerous queries can be `deny` or `approve`
`execute_write`	Submit a single `INSERT`, `UPDATE`, `DELETE` or DDL	`datasource_id`, `sql`, required `reason`	Management statements do not use this tool; DDL requires Agent to be `ddl`; `allow`/`warn` can only be executed. `approve` will automatically create a pending approval form but will not execute it.
`request_approval`	Take the initiative to submit SQL for review	`datasource_id`, `sql`, required `reason`	Only create approval form, do not execute SQL; return `approval_id`
`get_approval_result`	Check approval status	`approval_id`	Only the current Agent's own orders can be queried; the order numbers of other Agents are treated as not found.
`execute_approved`	Execute an approved archive request	`approval_id`	Single claim of the original Agent; recheck identity, data sources, policies and rules; reject repeated calls

`execute_write` and `request_approval` the caller `reason` It will be saved as an application description; the reasons for rule evaluation and approval notes will be saved separately.

v0.4.0's B5 cross-request logical session and PostgreSQL planned transactions are enabled by factory default, so by default `tools/list` will also return the following 8 tools:

Tools	Purpose
`open_session` / `close_session` / `get_session_status`	Create, close, and query explicit AgentSQL logical sessions; do not mix this with HTTP `Mcp-Session-Id` transport sessions
`begin_transaction`	Preflight and archive the complete ordered DML plan, then start the PostgreSQL transaction; no statements are executed at this step
`execute_transaction_statement`	Execute a sealed `INSERT`, `UPDATE` or `DELETE` according to plan ordinal. SQL cannot be replaced during the execution phase.
`commit_transaction` / `rollback_transaction`	Commit or rollback and complete the final state fence
`get_transaction_status`	Idempotent query transaction outcome, audit and disposition status

B5 still only allows one top-level SQL per operation, and does not support `SELECT`, `RETURNING`, DDL or MySQL cross-request transactions within a transaction. Explicitly setting `mcp.sessions.enabled: false` will hide these 8 tools and require `mcp.transactions.postgres` to be closed at the same time; the actual available set will always be based on `tools/list`.

Call timing

read-only query

```
list_datasources
  → list_schema
  → explain_query
  → query
```

First confirm the data source and visible structure, then evaluate, and finally execute. `explain_query` of `allow` It only means that the evaluation has passed, but it does not mean that the query has been completed.

Write operation: automatically transferred to approval

```
execute_write(reason required)
  → decision="approve" + approval_id (the system has automatically created a pending order)
  → The administrator approves in the console
  → get_approval_result
  → After approved, the original Agent calls execute_approved(approval_id)
```

After receiving `approval_id` returned by `execute_write`, do not call `request_approval` again, otherwise it will be submitted for review again.

`approved` will not automatically execute the original SQL. The original Agent can call `execute_approved` once within the validity period; the tool rechecks the current permissions and rules, and will close execution if it encounters rejection or failure. The old approval form cannot be executed and cannot be automatically retried after a single claim.

Write operation: actively submit for review

```
request_approval(reason required)
  → approval_id
  → The administrator approves in the console
  → get_approval_result
  → Call execute_approved(approval_id) after approval
```

Active submission itself never executes SQL; only `execute_approved` executes the archive request upon approval.

Correctly handle return values

- `decision:"allow"`: Allowed, the execution tool has been executed according to its semantics.
- `decision:"warn"`: Returns successfully with an alarm; execution tools can be executed, but warnings still need to be recorded.
- `decision:"deny"`: Policy or rule denial; is a tool result returned successfully, not a transmission error.
- `decision:"approve"`: Manual processing is required; SQL is not executed.
- `decision:"error"`: Processing failed; this is the only decision that sets MCP `IsError=true`.

Therefore, do not treat `deny`, `warn`, or `approve` as network errors with automatic retries. Especially for write operations, blind retry will result in repeated approvals or repeated business intentions.

Claude Desktop access

Claude Desktop use `claude_desktop_config.json` in `mcpServers` Start the native process.v1.0.0of `agentsql mcp` The standard MCP stdio line-by-line JSON message format is used by default; legacy custom length prefix clients can explicitly add `--sealed-stdio` Continue to use the original frame format. The following direct examples are applicable to desktop environments that can run AgentSQL Linux binaries; the target platform of this version's release assets is Linux amd64/arm64. The actual download must be based on the Release that has completed acceptance. For Windows desktops, please use the following HTTP bridging example. `command` It is recommended to fill in the absolute path of the executable file to avoid conflicts between desktop applications and terminals. `PATH` Inconsistent.

```
{
  "mcpServers": {
    "agentsql": {
      "command": "/absolute/path/to/agentsql",
      "args": ["mcp", "--config", "/absolute/path/config.yaml"],
      "env": {
        "AGENTSQL_SECRET": "<与控制面配对的恰好 32 字节值>",
        "AGENTSQL_API_KEY": "<Agent API Key>"
      }
    }
  }
}
```

The above parameters have been confirmed by the repository CLI implementation: the subcommand is `mcp` and the configuration parameter is `--config` (can be abbreviated to `-c`). After saving the configuration and restarting Claude Desktop, you should be able to see 12 basic tools and 8 B5 tools by default according to the tool registration rules (this version has not been tested in Claude Desktop, real client acceptance uses MCP Inspector and Dify); only basic tools are displayed when B5 is explicitly closed. If it does not appear, first confirm that the process launched by the desktop client can read the executable file, configuration file, metadata/audit control plane storage, and uses the same `AGENTSQL_SECRET` as the encrypted data source credentials.

Windows Claude Desktop: Connect to an existing AgentSQL service

First run the AgentSQL HTTP service on the local machine or the SSH local forwarding port to confirm that `http://127.0.0.1:7780/healthz` is normal. Windows clients can use [1] to convert local stdio to Streamable HTTP; it is a separate community bridge program and is not included in the AgentSQL release package. The following example uses the installed `mcp-remote`; please change `command` in the configuration file to the actual absolute path of the Node.js executable file, and change the first value of `args` to the absolute path of the actual `mcp-remote` entry script. Backslashes in Windows JSON paths need to be written `\\`.

```
{
  "mcpServers": {
    "agentsql": {
      "command": "C:\\path\\to\\node.exe",
      "args": [
        "D:\\path\\to\\mcp-remote\\dist\\proxy.js",
        "http://127.0.0.1:7780/mcp",
        "--transport", "http-only",
        "--header", "Authorization:${AGENTSQL_AUTH_HEADER}"
      ],
      "env": { "AGENTSQL_AUTH_HEADER": "Bearer <Agent API Key>" }
    }
  }
}
```

Here only one Agent API Key is given to the desktop bridging process. Do not use administrator tokens for MCP; do not expose listening ports to the public network. Claude's cloud custom connector initiates the connection from the Anthropic cloud network and cannot use the loopback address here; in this case, it is the Claude Desktop local process. The desktop client needs to complete its own login before it can view the tools on the interface. The protocol testing of this repository is not equal to the UI acceptance of the logged-in client.

MCP Inspector local acceptance (Streamable HTTP)

The old `/mcp` portal supports MCP `2025-06-18`. The accepted combination of this entrance is MCP Inspector `0.16.8` and its dependency `@modelcontextprotocol/sdk` `1.18.0`; the newer Inspector negotiates the update protocol by default and cannot directly apply this set of old entrance acceptance steps. Save the following `package.json` in a separate directory, run `npm install`, and then start the local `node_modules/.bin/mcp-inspector` (Windows PowerShell uses `./node_modules/.bin/mcp-inspector.cmd`):

```
{
  "private": true,
  "dependencies": {"@modelcontextprotocol/inspector": "0.16.8"},
  "overrides": {"@modelcontextprotocol/sdk": "1.18.0"}
}
```

First set `$env:HOST='127.0.0.1'` and `$env:ALLOWED_ORIGINS='http://127.0.0.1:6274'` in Windows PowerShell, then start Inspector and open the interface according to the local URL given by the terminal. Select Streamable HTTP for transmission, fill in `http://127.0.0.1:7780/mcp` for the address, and fill in the complete `Authorization: Bearer <Agent API Key>` in Authentication → Custom Headers. Do not add `MCP-Protocol-Version` manually, the SDK will automatically send it after initialization; repeated addition will form an invalid merge header. Check tool discovery, authorization query, unauthorized denial and corresponding audit ID in sequence. Use a separate test Agent Key, do not fill in the administrator token, and do not expose the Inspector agent port to the public network. Inspector is an MCP development testing tool and is not a replacement for AI Assistant or Claude Desktop's interface compatibility acceptance.

Dify self-hosted access (Streamable HTTP)

Add a new service in Dify's MCP tool management interface, fill in the Server URL with `http://<AgentSQL 内网地址>:7780/mcp` that is reachable by the Dify API container, and fill in both `Authorization: Bearer <Agent API Key>` and `MCP-Protocol-Version: 2025-06-18` in the request header. The MCP client of Dify 1.17.1 declares this version, but its HTTP transport does not automatically add the version header required by subsequent requests, so this must be configured explicitly. After saving, check the tool discovery first, and then call `list_datasources` and authorized read-only `query` from the workflow; when returning `deny`, `approve` or `error`, it will be handled according to the business decision and will not be regarded as a network exception that can be blindly retried. `127.0.0.1` in the Dify API container points to Dify itself and cannot point to the host AgentSQL.

Self-hosted Dify blocks access to private addresses through SSRF proxy by default; if AgentSQL is located on the same controlled intranet, only add the precise `/32` of AgentSQL's actual address to Dify's `SSRF_PROXY_ALLOW_PRIVATE_IPS`, and keep AgentSQL's external port controlled. Do not release the entire RFC1918 network segment for access testing. Dify's static MCP request header corresponds to a fixed Agent identity; each user's independent Agent identity requires a separate identity mapping design, and the Dify workspace identity cannot be automatically regarded as the AgentSQL identity.

Cursor access (stdio)

Cursor can put the same `mcpServers` fragment into user-level `~/.cursor/mcp.json`, or project-level `.cursor/mcp.json`. Team projects prefer to only submit configuration templates that do not contain secrets; the real Key and SECRET use the secret injection method provided by the current version of Cursor or are only saved in local configurations that are not included in version control.

```
{
  "mcpServers": {
    "agentsql": {
      "command": "/absolute/path/to/agentsql",
      "args": ["mcp", "--config", "/absolute/path/config.yaml"],
      "env": {
        "AGENTSQL_SECRET": "<与控制面配对的恰好 32 字节值>",
        "AGENTSQL_API_KEY": "<Agent API Key>"
      }
    }
  }
}
```

Relative paths in project-level files are easily affected by the working directory. It is recommended to use absolute paths for the values of `command` and `--config`. After saving, refresh the MCP Server according to the current Cursor version; if the configuration entry or outer fields of the client version change, the Cursor official MCP document shall prevail.

Doubao and other remote MCP clients (Streamable HTTP)

Clients must explicitly support **Streamable HTTP** and allow custom headers to be added to the MCP Server. Fill in the following on the MCP Server configuration page of Doubao or other clients:

Configuration items	Public network demonstration	Production self-hosted
Server URL	`https://demo.agentsql.cn/mcp`	`https://<你的网关>/mcp`
Header name	`Authorization`	`Authorization`

Configuration items	Public network demonstration	Production self-hosted
Header value	`Bearer <演示环境 Agent API Key>`	`Bearer <自托管 Agent API Key>`

The demo Key is the Agent API Key, not the administrator token of the demo console; when using the demo administrator account prefilled on the login page, you should confirm the current read-only Key corresponding to the selected demo Agent. If the demo key is not public, use self-hosted environment verification instead, and do not guess or reuse administrator tokens. The demo environment is for local access testing only, production must be self-hosted and use its own keys. If a client does not support custom headers, only supports older versions of SSE, or its Streamable HTTP version cannot be confirmed, it cannot be directly connected in this way; do not put the Key in the URL query parameter.

The following is the general field skeleton, the configuration entry of Doubao and other clients, the outer JSON structure, and the `type` or `transport` field required by some versions. The current official documentation of the client shall prevail:

```
{
  "mcpServers": {
    "agentsql": {
      "url": "https://demo.agentsql.cn/mcp",
      "headers": {
        "Authorization": "Bearer <Agent API Key>"
      }
    }
  }
}
```

When the native AgentSQL service is already running, you can also change the URL to `http://127.0.0.1:7780/mcp`. Non-native production deployment must use a controlled HTTPS entrance, and the listening port must not be exposed directly to the public network.

- Cursor: When accessing remotely, use the official MCP document of this version to confirm the HTTP transport field.
- Claude Desktop: Use the official MCP document to confirm whether this version supports Streamable HTTP; if it only supports stdio, use the previous Claude Desktop configuration.
- Cline: Use the official MCP document to confirm the configuration entry, `mcpServers` structure and HTTP transport field.

The `command`, `args`, `env` (stdio) and `url`, `headers` (HTTP) of these clients are the information required for AgentSQL connection; the outer structure and transport statement are subject to the current version of the client.

Find from MCP Registry / Glama

AgentSQL has been included in the MCP official Registry (status is active). Search for **AgentSQL** in the MCP Registry, check the entry name `io.github.cuipengdba/agentsql`, version and repository address, and then access it according to the method supported by the client; the repository does not provide a direct URL to the Registry entry that can be relied on for a long time, so this article does not make up links.

AgentSQL has also been included in Glama and is available atAgentSQL search results for GlamaFind in . Endpoints, versions, or header examples in third-party directories may lag, ultimately resulting in this repository documentation, actual deployment configuration, and MCP `tools/list` Subject to return.

Client access security reminder

- The public network demo only connects to the synthetic demonstration library for functional experience and client connectivity testing; do not send real business SQL, data or credentials to the demo.
- The production environment must self-host AgentSQL and use its own data source, Agent and API Key; configure TLS, access control and current limiting in front of the gateway.
- API Key, `AGENTSQL_SECRET`, and data source password must not be written into public configurations, screenshots, logs, or code repositories. Only placeholders are kept in public projects and rotated immediately after leaks.
- The stdio configuration will create a new AgentSQL runtime on the local machine; if there is already a controlled service, its Streamable HTTP endpoint will be connected first to avoid copying or relaxing production secret file permissions.
- Clients should only obtain the minimum Agent capabilities and data source authorization required to complete the task; always handle `deny`, `warn`, `approve`, `error`, and do not treat security decisions as network failures and automatically retry.

Self-developed client

initialize naked HTTP example

Set the Key first; the following request applies to the Streamable HTTP service that is already running on this machine:

```
# Applicable premise: the local AgentSQL is running; AGENTSQL_API_KEY has been exported as a valid Agent
Key
curl -fsS -D /tmp/agentsql-mcp-headers http://127.0.0.1:7780/mcp \
  -H "Authorization: Bearer ${AGENTSQL_API_KEY}" \
  -H 'Content-Type: application/json' \
  -H 'Accept: application/json, text/event-stream' \
  --data '{"jsonrpc":"2.0","id":1,"method":"initialize","params":{"protocolVersion":"2025-06-
18","capabilities":{},"clientInfo":{"name":"agentsql-smoke","version":"1.0.0"}}}'
```

Streamable HTTP transports are stateful by default. Read `Mcp-Session-Id` in the initialize response and carry it unchanged in subsequent `tools/list`, `tools/call`, GET and DELETE requests. The first initialize negotiates the version via `params.protocolVersion`, so its HTTP protocol version header can be omitted; subsequent POSTs must send `MCP-Protocol-Version: 2025-06-18`. The server will still re-verify the Bearer for each request. `mcp.http.stateful: false` is only used for compatibility fallback, in which session headers will not be returned and only POST will be accepted. Regardless of mode, each request must send a Bearer and the correct `Accept`, and keep the request body within 4 MiB.

Go SDK

It is recommended to use `StreamableClientTransport` of `github.com/modelcontextprotocol/go-sdk` and inject it for each request by customizing `http.RoundTripper`:

```
Authorization: Bearer <Agent API Key>
MCP-Protocol-Version: 2025-06-18
Accept: application/json, text/event-stream
```

Do not output the complete header in the log, and do not treat `deny` as a transport error.

Troubleshooting

`401 unauthorized`

Check whether the administrator token has been misused or the Agent Key is incorrect. After the Agent is disabled, expires, deleted, or rotated, the old Key will become invalid; rotation will not retain the concurrent validity period.

stdio has no console UI

This is expected behavior. stdio will close `console_enabled` and the event stream. Installed services should use HTTP so that the console continues to be served by the systemd running instance.

Can only be accessed locally on the server

The default loopback binding is safe by design. Use SSH `-L`, or access after controlled TLS reversion; do not change to the public network `0.0.0.0` for streaking.

MCP handshake failed

Confirm that the endpoint is `/mcp` and the request contains `Content-Type: application/json`, `Accept: application/json, text/event-stream`, and `MCP-Protocol-Version: 2025-06-18`. In the default stateful mode, also confirm that the initialize response contains `Mcp-Session-Id`, and subsequent requests carry the same value; also check whether the client really supports Streamable HTTP, and whether the request body exceeds 4 MiB.

The new entry `/mcp/2026-07-28` does not require a handshake, and each request must carry the version header and `params._meta`; `server/discover` only advertises `2026-07-28`. Missing, old, or conflicting version metadata returns HTTP 400, oversized request bodies return 413, and cross-site Origin returns 403. For more protocol rejection and upgrade troubleshooting, see current version operation and maintenance manual .

Stacked multiple statements are rejected

This is the expected behavior of R001. Send only one SQL statement at a time; do not append a second statement to a legitimate query.

Write operation prompt is missing `reason`

Both `execute_write` and `request_approval` require non-empty `reason`. This type of early rejection at the tool layer occurs before entering the pipeline and may not have an audit record; resubmit after supplementing the purpose, scope of impact, and rollback plan.

stdio cannot connect to the control plane or cannot decrypt the data source

Check that the process can read the configuration and SQLite/PG control plane and is using the same `AGENTSQL_SECRET` paired with the data source ciphertext. The installer's `/etc/agentsql/agentsql.env` is `0600 root:root`; do not copy or loosen the production secret file if it cannot be read by regular desktop users, use the HTTP endpoint of the running service instead.

SQLBot

The v2 adapter in v3.2.0 maps each ordinary SQLBot v1.10.2 login user, workspace, and datasource to an AgentSQL person and read-only Agent. Administrators can configure bindings, export configuration without secrets, rotate connector credentials, and inspect configuration diagnostics in the console. The trusted SQLBot server supplies the current user identity; every call rechecks the binding and current permissions. Missing bindings or identity cause rejection.

Personal bindings cover specified PostgreSQL and MySQL read-only scenarios, pinned source hashes, and controlled SQL subsets. They include model metadata filtering, structured denial references, and read-only approvals explicitly initiated by users. For MySQL, the discovery schema must match the datasource database name. SQLBot providing its own MCP Server does not mean it calls AgentSQL by default. See the SQLBot Integration Guide for installation, independent Agent Key configuration, TLS, the approval console, and legacy v1 migration boundaries.